Institute of Mathematics and Computer Science
European Bioinformatics Institute

SIMS
Installation Guide

Version 2.14

Pre-requirements

To install Sample Information Management System the following software needs to be preinstalled on your system:

- **Apache Tomcat** suitable for your OS. Tomcat version 6.0 or higher is recommended, although in principle Tomcat versions 5.x *might* work. Tomcat installation will also require a JRE compatible with a particular Tomcat version (JDK is required if Tomcat versions 5.x is used).
- **PostgreSQL** database. The current version of SIMS is tested with PostgreSQL version 8.4 and use of this version is recommended. Alternatively, you can also use another relational database of your choice, however in this case it is likely that you will need to make some modifications in database initialization scripts. In principle SIMS has been successfully tested on MySQL and Oracle databases.

Installation procedure

SIMS installation requires the following steps:

- Creation of SIMS database schema. Since several SIMS instances could be run on the same server, it is assumed that an empty PostgreSQL database has already been created and initialization script will create the required tables in already existing database. There are at least two simple ways to use initialization script:
    - if you use **pgAdmin** tool for PostgreSQL management, select the schema of the currently empty database where you wish to install SIMS (by default this will be called *public*), open *Execute arbitrary SQL queries* window, paste the contents of installation script here and press *Execute query*.
    - insert the line `\connect <your database name>` as the first line in installation script and run the command `psql –U postgres –f <name of initialization script file>`. You may need either to place script file in a directory visible to `psql`, or give the full path of initialization script file.
- Deployment of SIMS software. The simplest way to do this is to unzip Tomcat deployment file in directory $CATALINA_HOME/webapps ($CATALINA_HOME will be defined by your Tomcat installation). The name of deployment directory can be freely chosen and/or changed. The SIMS then most likely will be accessible via web addresses **http://your.server.name:8080/<your SIMS directory>** (if Tomcat is installed on default port 8080) or **http://your.server.name/<your SIMS directory>** (if Tomcat is installed on port 80). Other ports can also be used, but if the SIMS is intended to be accessed remotely the use of standard ports 80 or 443 (if you wish to use secure HTTPS protocol for accessing SIMS).
  The other alternative is to recompile the software (for this you will need both Tomcat deployment and Java sources archives).
- SIMS configuration. Currently all configuration settings are accessible in configuration files WEB-INF/web.xml and conf/systemConst.xm located in directory in which you deployed your SIMS application. The contents of these files are described in details in a separate section of this guide.
  In addition you might want (or need) to change some settings in file WEB-INF/classes/hibernate.properties (this will need to be modified if you use other database engine instead of PostgreSQL) and/or other .properties files in the same directory.
- Initialization of SIMS instance. This includes two tasks:
    - To access SIMS you need a login name and password. If no users are defined as yet (table *Users* is empty) a new user will be created the first time when you attempt to log in the system. The *login name and password for this user will be those which you will provide for this first login* (thus it is important not to lose them!), the informal description of this user will be *First user* and the user will be given full administrator rights. Also one user group, called *default group* will be created and the first user will be made member of this user group. If for some reason you lose the login name and/or password to access your SIMS instance, a possible solution is to delete all entries from tables *Users* and *User groups*, the *First user* then will be recreated when you will try to access the system.
    - For SIMS to work correctly all metadata fields must be configured in tables *Person metadata*, *Sample metadata* and *Aliquot metadata*. In current version this task is simple: in **AdminTables** click on link I*mport metadata and vocabularies*, select the right *Metadata file* for your SIMS version (the file should be in XML format – unzip it first) and click on *Import* button. This will fill all metadata files, as well put entries in controlled vocabularies (if they are used). Re-importing the same file in already initialized system will have no

effect. If metadata configurations are incomplete or incorrect, the application will give an error and the data will not be imported.

It is possible to install several SIMS databases on PostgreSQL and it is possible to install several SIMS instances on Tomcat.

SIMS configuration

The configurable parts of WEB-INF/web.xml file are described below:

Specify URL to your SIMS database. This should include database location (in example below it is assumed that PostgreSQL is being run on the same server as Tomcat) and your database name (it example below it is assumed that database name is *sims*).

```
<context-param>
  <param-name>sample.url</param-name>
  <param-value>jdbc:postgresql://localhost/sims</param-value>
</context-param>
```

Specify user name to access database. The example below assumes that it is *postgres*, but since this is usually the default PostgreSQL user with administrator rights, creation and configuration of a special user to access SIMS is recommended.

```
<context-param>
  <param-name>sample.username</param-name>
  <param-value>postgres</param-value>
</context-param>
```

Specify password for this database user. The example below assumes that it is *my_password*, but use of more complicated passwords is recommended.

```
<context-param>
  <param-name>sample.password</param-name>
   <param-value>my_password</param-value>
</context-param>
```

Number of entries that will be shown in a single page of **List of <level>** (usually top level is “Persons”, middle is “Samples” or “Visits” and bottom level is “Aliquots” or “Samples”). Chose a value that best suits your needs.

```
<context-param>
  <param-name>viewlimit</param-name>
  <param-value>1000</param-value>
</context-param>
```

Specify the directory where data files will be stored. The example below assumes that directory is `/sims/data_files`. AS above, it is it is strongly recommended to use full path.
Don’t place this directory within $CATALINA_HOME/webapps – you may lose all your data files, if by accident you ask Tomcat to undeploy your SIMS instance.

```
<context-param>
  <param-name>data.directory</param-name>
  <param-value>/sims/data_files</param-value>
</context-param>
```

Specify the directory where supplementary files will be stored. The example below assumes that directory is `/sims/supplementary_files`. AS above, it is it is strongly recommended to use full path.
Don’t place this directory within $CATALINA_HOME/webapps – you may lose all your supplementary files, if by accident you ask Tomcat to undeploy your SIMS instance.
Directories for data and supplementary files have to be separate.

```
<context-param>
  <param-name>protocol.directory</param-name>
  <param-value>/sims/supplementary_files</param-value>
</context-param>
```

If the application is configured to work with an ftp server to use for file import working directory must be provided. You can uncomment and configure the following two parameters, if option of file upload via FTP is enabled. In Examples below it is assumed that ftp server name is `my.ftp.server.com`.

```
<context-param>
  <param-name>ftp.server</param-name>
  <param-value>my.ftp.server.com</param-value>
</context-param>
<context-param>
  <param-name>ftp.directory</param-name>
  <param-value>ftp/server/directory</param-value>
</context-param>
```

For the users convenience it is also advisable to configure the informative popup information. These values are used only in pop-up tip **Batch upload** pages to remind user about the available FTP account for data upload. You are likely to benefit from this option if a single account for FTP upload is shared between all users.

```
<context-param>
  <param-name>ftp.directory.popup</param-name>
  <param-value>ftp/server/directory</param-value>
</context-param>
<context-param>
  <param-name>ftp.server.popup</param-name>
  <param-value>my.ftp.server.com</param-value>
</context-param>
```

System Configuration file

There are several constants that need to be specified. These are kept in **conf/systemConst.xml** file. They are not server specific but are instead adjusted to each configuration (e.g. MS, AD or T2D). **If one of the default configurations are used with no alterations the following can be skipped** and the file used with no change.

The file begins with an opening tag:

```
<SIMS_Constants version="1.17">
```

There are upload names for static fields:

```
<UploadNames>
<UploadName column="visibleName" upload="ID"/>
<UploadName column="comment" upload="COMMENT"/>
<UploadName column="dataFiles" upload="DATA_F"/>
…
<UploadName column="modifier" upload=""/>
</UploadNames>
```

This is where you would change the level names of the application. Below each of the names of the three levels of the application are the numbers of each type of fields in that level. You should check that the number of entries in the corresponding level mapping file match the ones provided in this file:

```
<MainNames>
<Bottom plural="Aliquots" singular="Aliquot" prefix="Aliquot."
  string_numb="1" integer_numb="0" float_numb="2" date_numb="2"/>
<Middle plural="Characteristics" singular="Characteristic"
prefix="Characteristic."
  string_numb="3" integer_numb="0" float_numb="0" date_numb="1"/>
<Top plural="Persons" singular="Person" prefix="Person."
  string_numb="2" integer_numb="6" float_numb="4" date_numb="5"/>
</MainNames>
```

There is also a list of all the vocabularies. The seq_numb value must correspond to the value of VTxx.hbm.xml file of the corresponding vocabulary. The table and item values are going to be used in the application.

```
<Vocabularies>
  <Vocabulary seq_numb="01" table="Aliquot Concentration units"
item="Concentration unit"/>
 …
  <Vocabulary seq_numb="29" table="Sample subtypes" item="Sample
subtype"/>
  <Vocabulary seq_numb="30" table="Sample transport conditions"
item="Sample transport condition"/>
</Vocabularies>
```

The tables that will influence the generation of IDs are configured separately. First there is a list of the table mappings similar to that of vocabularies above.

```
<Identificies>
<Identificy seq_numb="1" table="Sample timings" item="Sample timing"/>
</Identificies>
```

Then the desired structure of the generated ID is defined. The level of the ID in question is defined, static strings (s1, s2) are defined, which table to be used (ivt1), and what counter (numbers or letters) (ptype).

```
  <IdStructure>
    <Middle s1="-" ivt1="1" s2="" ptype="1"/>
  </IdStructure>
```

Table dependencies are configured as follows. Again, the seq_numb corresponds to the number of DVTxx.hbm.xml file. The name of the dependency table, independent and dependent will be used in the application. And the names of the mapping files of the related vocabularies have to be provided.

In this example the independent table is the same one that participates in ID generation. It is not mandatory; the independent table can be any of the other vocabularies, as linked in the data base.

```
<Dependencies>
<Dependency seq_numb="1" table="Sample type and subtype"
independent="Type" dependent="Subtype"
                    independent_class="IVT01" dependent_class="VT16"/>
</Dependencies>
```

The file ends with a closing tag.

```
</SIMS_Constants>
```

This file is read by the application only once at startup. For any changes thereafter to take place the application needs to be reloaded by Tomcat server.